Select .net framework and select empty and check webapi checkbox

Install this

* Microsoft.Owin.Host.SystemWeb
* Microsoft.Owin.Security.OAuth
* Microsoft.Owin.Cors

**Create APIAUTHORIZATIONSERVERPROVIDER.cs Class File**

1. **public** **class** APIAUTHORIZATIONSERVERPROVIDER : OAuthAuthorizationServerProvider
2. {
3. **public** override async Task ValidateClientAuthentication(OAuthValidateClientAuthenticationContext context)
4. {
5. context.Validated(); //
6. }
7. **public** override async Task GrantResourceOwnerCredentials(OAuthGrantResourceOwnerCredentialsContext context)
8. {
9. var identity = **new** ClaimsIdentity(context.Options.AuthenticationType);
10. **if** (context.UserName == "admin" && context.Password == "admin")
11. {
12. identity.AddClaim(**new** Claim(ClaimTypes.Role, "admin"));
13. identity.AddClaim(**new** Claim("username", "admin"));
14. identity.AddClaim(**new** Claim(ClaimTypes.Name, "Hi Admin"));
15. context.Validated(identity);
16. }
17. **else** **if** (context.UserName == "user" && context.Password == "user")
18. {
19. identity.AddClaim(**new** Claim(ClaimTypes.Role, "user"));
20. identity.AddClaim(**new** Claim("username", "user"));
21. identity.AddClaim(**new** Claim(ClaimTypes.Name, "Hi User"));
22. context.Validated(identity);
23. }
24. **else**
25. {
26. context.SetError("invalid\_grant", "Provided username and password is incorrect");
27. **return**;
28. }
29. }
30. }

**Create a AuthenticationStartup.cs Class File**

1. **public** **class** AuthenticationStartup
2. {
3. **public** **void** Configuration(IAppBuilder app)
4. {
5. app.UseCors(Microsoft.Owin.Cors.CorsOptions.AllowAll);
6. var myProvider = **new** APIAUTHORIZATIONSERVERPROVIDER();
7. OAuthAuthorizationServerOptions options = **new** OAuthAuthorizationServerOptions
8. {
9. AllowInsecureHttp = **true**,
10. TokenEndpointPath = **new** PathString("/token"),
11. AccessTokenExpireTimeSpan = TimeSpan.FromDays(1),
12. Provider = myProvider
13. };
14. app.UseOAuthAuthorizationServer(options);
15. app.UseOAuthBearerAuthentication(**new** OAuthBearerAuthenticationOptions());
16. HttpConfiguration config = **new** HttpConfiguration();
17. WebApiConfig.Register(config);
18. }
19. }

**Create a APIAUTHORIZEATTRIBUTE.cs Class File**

1. **public** **class** APIAUTHORIZEATTRIBUTE : System.Web.Http.AuthorizeAttribute
2. {
3. **protected** override **void** HandleUnauthorizedRequest(System.Web.Http.Controllers.HttpActionContext actionContext)
4. {
5. **if** (!HttpContext.Current.User.Identity.IsAuthenticated)
6. {
7. base.HandleUnauthorizedRequest(actionContext);
8. }
9. **else**
10. {
11. actionContext.Response = **new** System.Net.Http.HttpResponseMessage(System.Net.HttpStatusCode.Forbidden);
12. }
13. }
14. }

**Create a controller with name UserController**

1. [AllowAnonymous]
2. [HttpGet]
3. [Route("api/data/forall")]
4. **public** IHttpActionResult Get()
5. {
6. **return** Ok("Now server time is: " + DateTime.Now.ToString());
7. }
8. [Authorize]
9. [HttpGet]
10. [Route("api/data/authenticate")]
11. **public** IHttpActionResult GetForAuthenticate()
12. {
13. var identity = (ClaimsIdentity)User.Identity;
14. **return** Ok("Hello " + identity.Name);
15. }
16. [Authorize(Roles = "admin")]
17. [HttpGet]
18. [Route("api/data/authorize")]
19. **public** IHttpActionResult GetForAdmin()
20. {
21. var identity = (ClaimsIdentity)User.Identity;
22. var roles = identity.Claims
23. .Where(c => c.Type == ClaimTypes.Role)
24. .Select(c => c.Value);
25. **return** Ok("Hello " + identity.Name + " Role: " + string.Join(",", roles.ToList()));
26. }

Addd this line in Startup file at top of namespace

[assembly: OwinStartup(typeof(DotnetWebAPI.AuthenticationStartup))]















